**Trabajo práctico Nro. 1**

|  |  |  |
| --- | --- | --- |
|  | **Asignatura: Programación II** |  |
|  |  |
| **Cursado:** Segundo semestre | **Horas** **semanales**: |
|  | **Horas semestrales:**  *Cantidad estimada de horas semestrales/anuales.* |
| **Carrera**: *Tecnicatura Universitaria en*  *Programación* | **Nivel (Año):**    3°  1  °  2  ° |
| **Ciclo Lectivo: 2024** |

**Integrantes de la Cátedra:**

* **DOCENTES:**

|  |  |  |
| --- | --- | --- |
| **Nombre del Profesor** | **Periodo** | **Cantidad horas materia** |
| **Cinthia Rigoni** |  | 6 horas |

* 1. ¿Qué es un TAD?

Los Tipos de Datos Abstractos proporcionan una forma de modelar objetos del mundo real en un programa informático, permitiendo una organización más clara y modular del código. Definen un conjunto de valores y un conjunto de operaciones que se pueden realizar sobre esos valores, pero ocultan los detalles internos de cómo se implementan esas operaciones, por eso se dice que son abstractos.

* 1. ¿Dónde se produce el encapsulamiento?

El encapsulamiento se produce en las clases.

* 1. ¿Cuáles son las semejanzas y las diferencias entre funciones, procedimientos y métodos?

Semejanzas:

* Los tres son bloques de código que realizan una tarea específica.

Diferencias:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Funciones** | **Procedimientos** | **Métodos** |
| **Devuelve un valor** | Si | No | Puede devolver un valor o ser void. |
| **Está específicamente asociada a un objeto** | No, pero puede estar dentro de una clase | No, pero puede estar dentro de una clase | Si |
| **Desde donde puede ser llamada** | Desde cualquier parte del código siempre que estén visibles en el ámbito en el que se utilizan | Desde cualquier parte del código siempre que estén visibles en el ámbito en el que se utilizan | Se invocan en el contexto de una instancia (objeto) de la clase en la que están definidos. |
| **Acceso a datos internos del objeto** | No, a menos que se les pase explícitamente como argumentos | No, a menos que se les pase explícitamente como argumentos | Si |

* 1. ¿Qué es UML? ¿Y cómo se representa una clase en ese lenguaje?

El Lenguaje Unificado de Modelado (Unified Modeling Language) es un lenguaje gráfico que se utiliza para representar los diferentes elementos que componen un sistema de software, como clases, objetos, relaciones, procesos y estados.

Una clase se representa en UML como un rectángulo dividido en tres partes: nombre, atributos y métodos. Además podemos ver la accesibilidad de los atributos y de los métodos con símbolos: + public, - private, # protected.

* 1. Marcar con cruz.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | No se aplica a clases | Solo se aplica a atributos | Solo se aplica a clases | Se aplica a atributos, métodos y clases |
| Public |  |  |  | **X** |
| Private | **X** |  |  |  |
| Protected | **X** |  |  |  |
| Static | **X** |  |  |  |
| Final |  |  |  | **X** |
| Primera letra en minúscula | **X** |  |  |  |
| Primera letra en mayúscula |  |  | **X** |  |

* 1. Verdadero o falso

* Un constructor…
* Es el método principal para ejecutar un programa. **F**
* Crea instancias. **V**
* Devuelve el valor de un atributo privado. **F**
* Tiene sentencia return. **F**
* Siempre existe uno por defecto, sin parámetros ni inicializaciones de atributos. **V**
* Se puede sobreescribir. **V**
* Se puede sobrecargar. **V**
* Su nombre se escribe con mayúscula. **V**
* Su calificador de acceso es static. **F**
* Su tipo de devolución no se indica y corresponde a la clase. **V**
* Un método…
* Puede tener múltiples parámetros con el mismo nombre, siempre y cuando tengan tipos diferentes. **F**
* Puede sobrecargarse. **V**
* Puede sobreescribirse. **V**
* Puede ser static. **V**
* Puede ser tanto public como protected, pero no private. **F**
* Un método puede tener un modificador de acceso final. **V**

7. Calificadores de acceso. Completa.

1. Se necesita que cualquiera pueda acceder al color de un vehículo. Entonces, declaro color como: public

b- Se necesita que color se pueda acceder a través no sólo de vehículo, sí no ahora también de Buses, y como todos sabemos un bus es un tipo de vehículo, entonces también deberá tener acceso a color. Entonces, declaro color como: protected

c- Se necesita que color se pueda acceder solamente para vehículo. Entonces, declaro color como: private